



---

# *iWarp-Protocol Kernel-Space Software Implementation*

Dennis Dalessandro, Ananth Devulapalli, Pete Wyckoff

{dennis, ananth, pw}@osc.edu

*Ohio Supercomputer Center*

---

# Overview

---

- Introduction
- Motivation: *Why Software iWarp?*
- iWarp: Details
- Design Issues
- Experiments & Results
- Our future goals

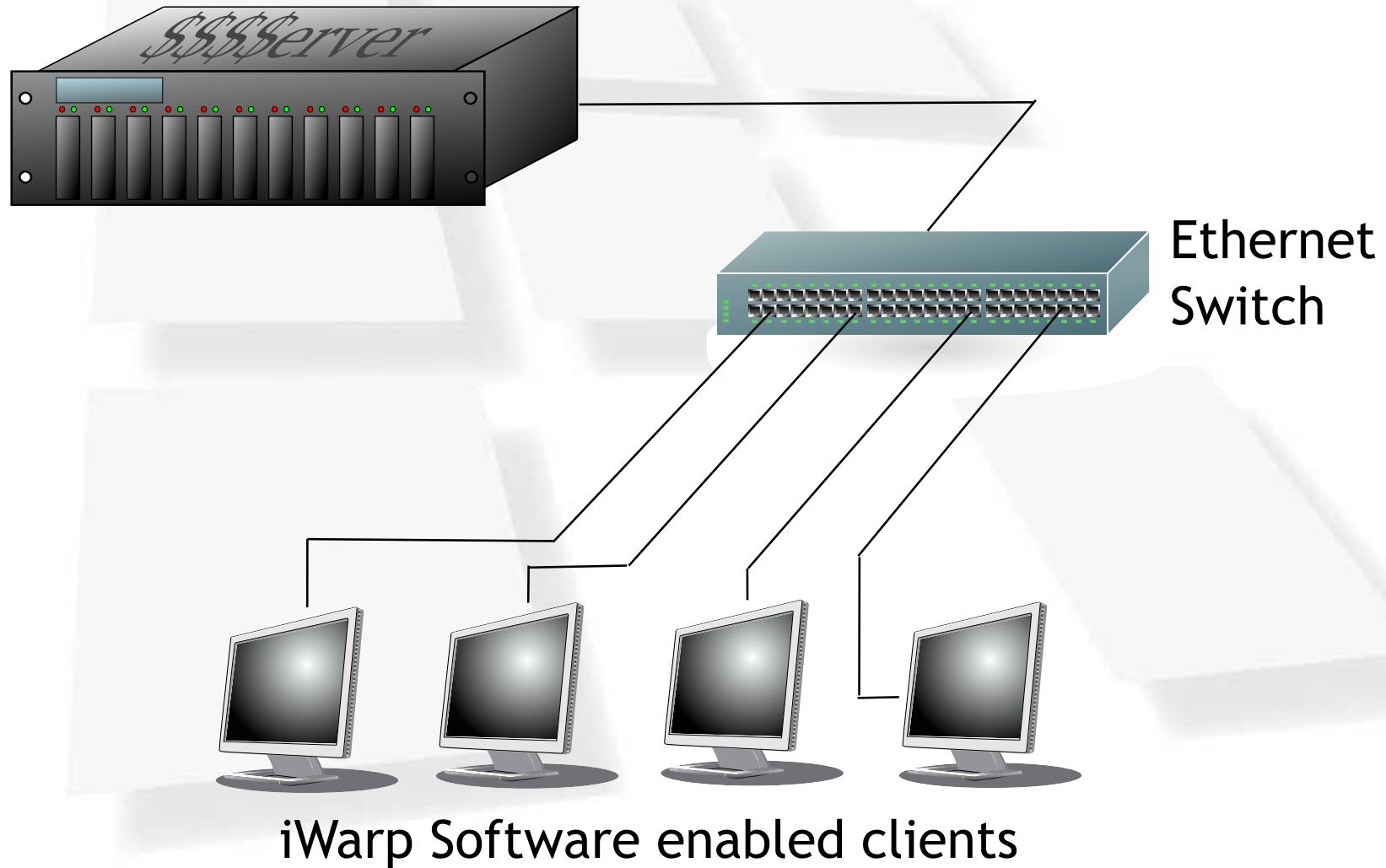
# Introduction

---

- High Performance Interconnects
  - ◆ Zero-copy
  - ◆ RDMA
  - ◆ Specialty protocol
  - ◆ LAN-wide
- RDMA over Ethernet → *iWarp*
  - ◆ Reduce end-point bottleneck
  - ◆ 10 Gbps at 3-4 GHz

# Motivation

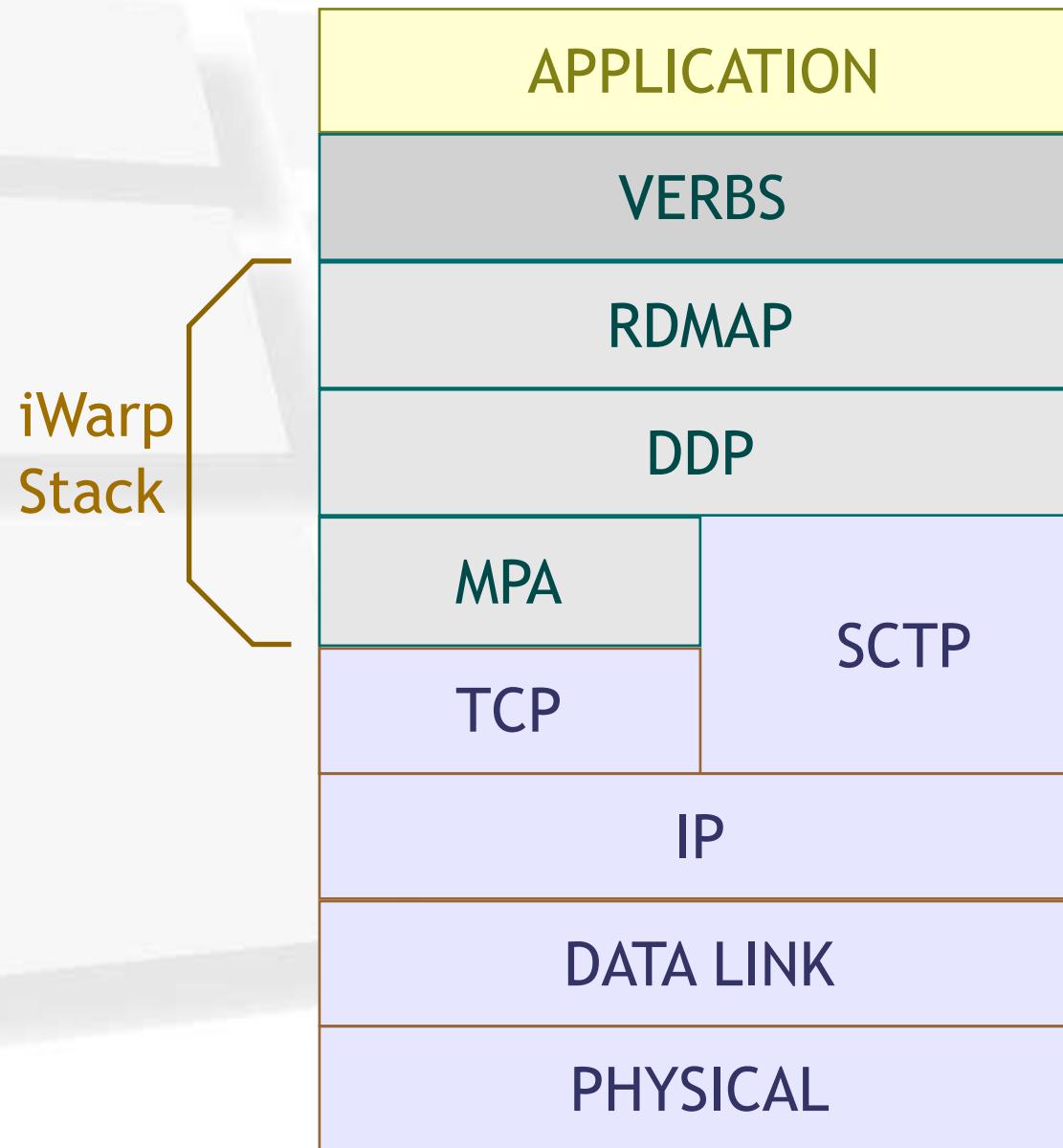
iWarp Hardware enabled Server



# Motivation (contd.)

- Flexible Research Platform
  - ◆ Protocol Experimentation
  - ◆ Protocol Compliance
  - ◆ Extensible to other protocols: *iSER*, *SRP*
- Advantages of iWarp in kernel
  - ◆ Unlock iWarp for kernel-resident clients: *NFS*
  - ◆ Coupling with TCP
  - ◆ Reduction in overhead

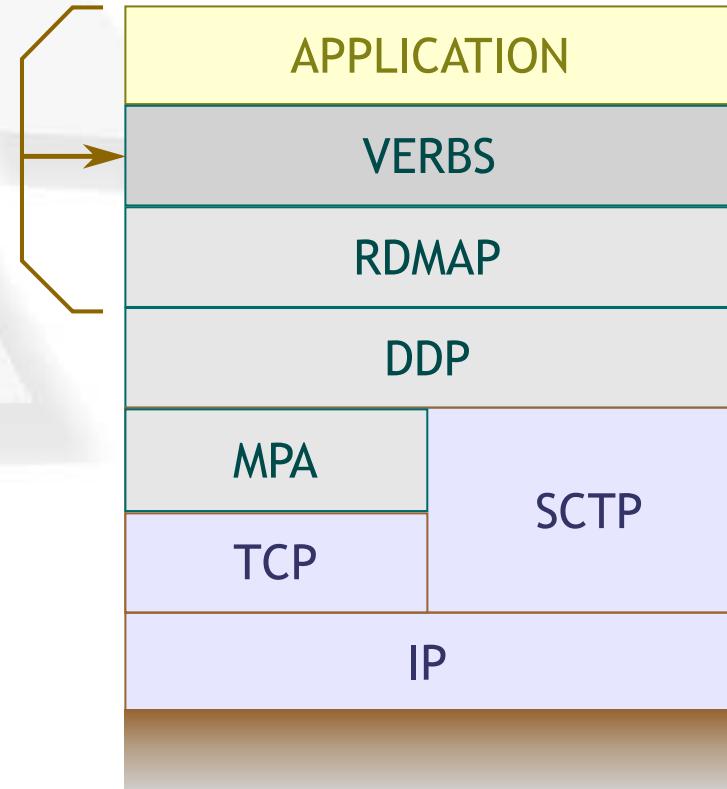
# iWarp Details



# iWarp Verbs

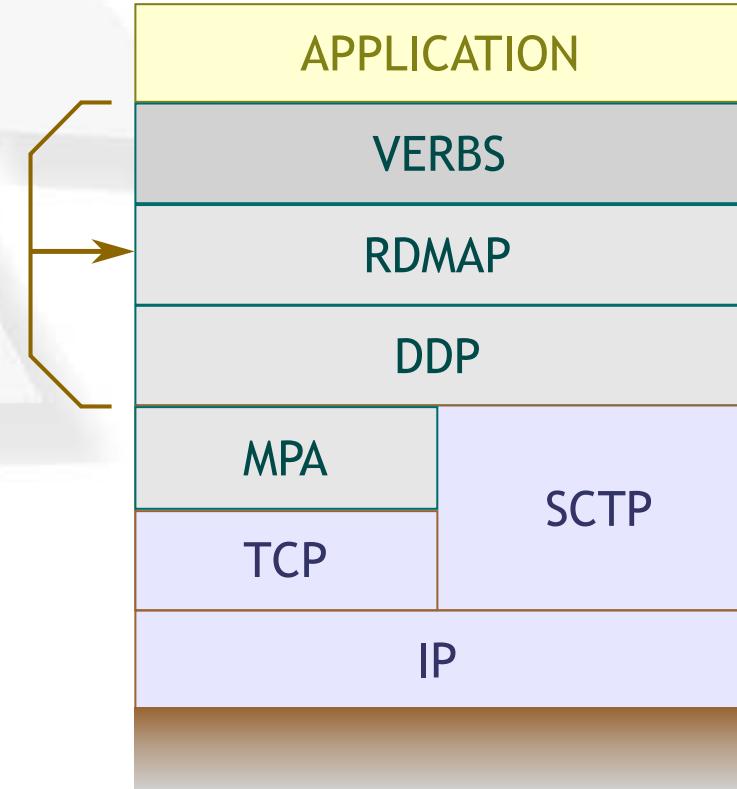
- Interface to application
  - ◆ Unlocks full power of iWarp unlike *Sockets semantics*
- Inspired by IB Verbs
  - ◆ QPs, Work Descriptors etc.

<http://www.rdmaconsortium.org/home/draft-hilland-iwarp-verbs-v1.0-RDMAC.pdf>



# RDMA Protocol Layer (RDMAP)

- Communication oriented layer
- Exports high level transport services to ULP
  - ◆ RDMA Write, RDMA read, Send, Recv
- Resource bookkeeping
  - ◆ Socket/streams
- Buffer management jointly with DDP
  - ◆ Informs message destination to lower layers
- Error surfacing to ULP (Verbs)

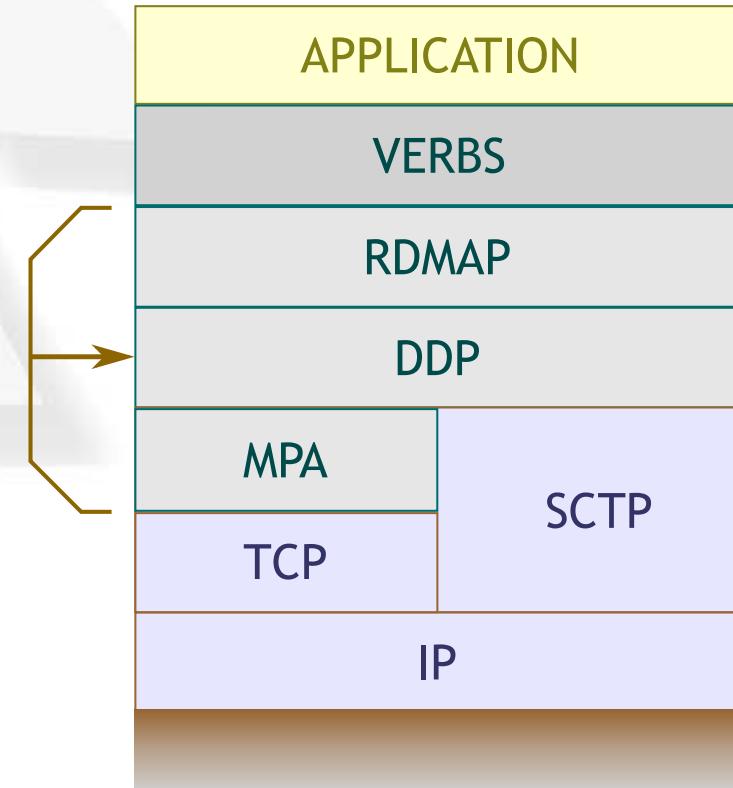


<http://www.ietf.org/internet-drafts/draft-ietf-rddp-rdmap-04.txt>

# Direct Data Placement (DDP)

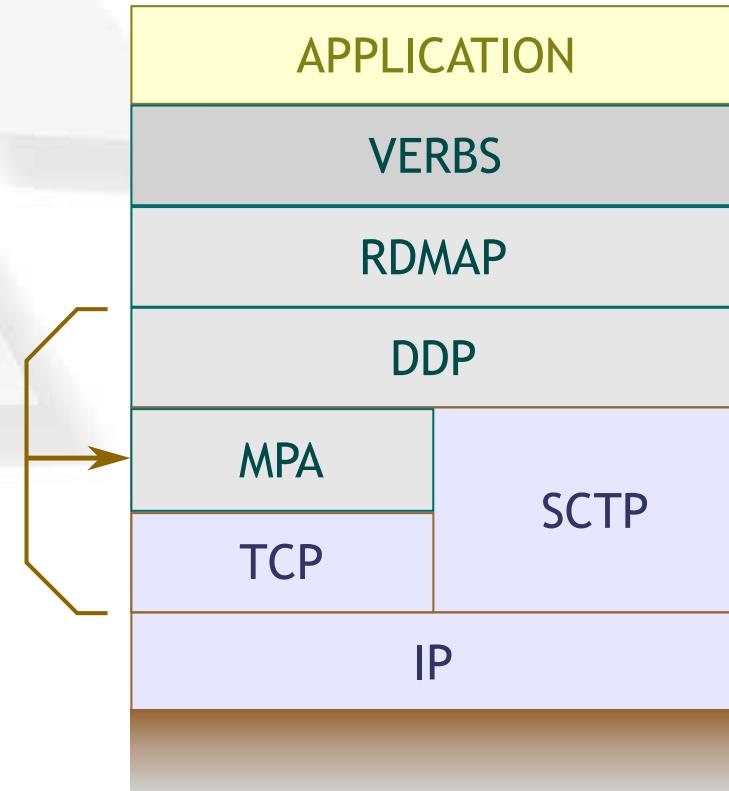
- Message oriented layer
- Zero-copy data placement
- Two data transfer models
  - ◆ Tagged
  - ◆ Untagged
- Segmentation/Re-assembly
  - ◆ Segment size from LLP (MPA)
- Out-of-order placement but In-order delivery

<http://www.ietf.org/internet-drafts/draft-ietf-rddp-ddp-04.txt>



# Marker PDU Aligned framing (MPA)

- Segment/Packet oriented layer
- Why MPA?
  - ◆ DDP's requirement of record boundaries
  - ◆ MPA + TCP or SCTP
- Markers for Framing
- Message integrity: CRC-32C
- Startup phase
  - ◆ Marker, CRC negotiation



<http://www.ietf.org/internet-drafts/draft-ietf-rddp-mpa-02.txt>

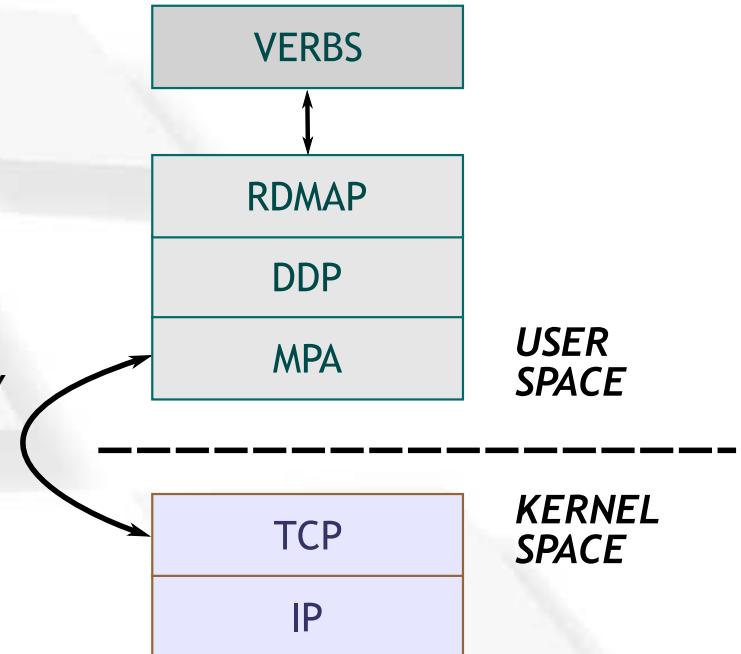
# Design Issues

---

- Verbs
- MPA ↔ TCP Interface
- Threading Model
- User ↔ Kernel Interface

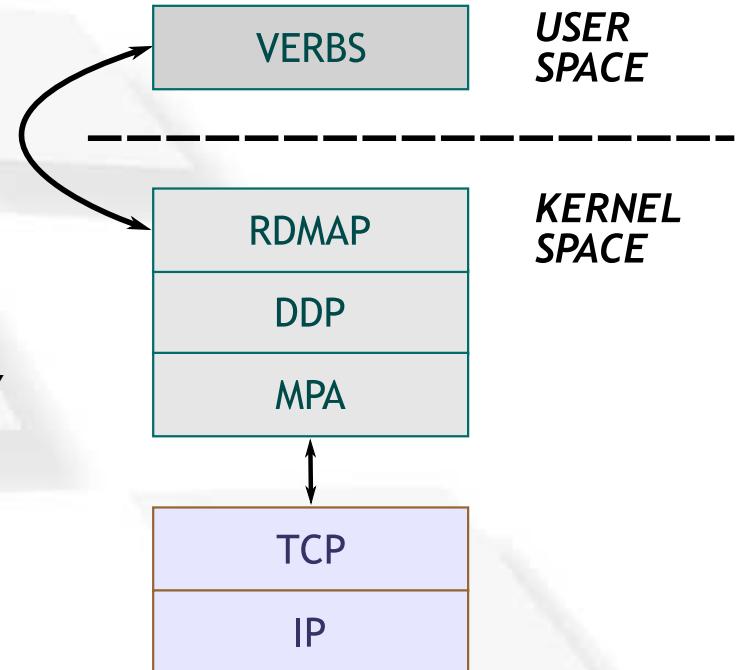
# Design Issues: Verbs

- Verbs or API like DAPL?
- User-space resident
- Modularized implementation
  - ◆ Code reuse: User-space and kernel-space
- *Minimize scope without sacrificing functionality*



# Design Issues: Verbs

- Verbs or API like DAPL?
- User-space resident
- Modularized implementation
  - ◆ Code reuse: User-space and kernel-space
- *Minimize scope without sacrificing functionality*



# Design Issues: MPA-TCP Interface

- MPA and TCP tightly Coupled
  - ◆ Control over TCP behavior ☺
  - ◆ Control over frames ☺
  - ◆ Portability ☹
- MPA and TCP loosely Coupled
  - ◆ `kernel_sendmsg`, `kernel_recvmsg`
    - Blocked sends
    - Polling recvs
  - ◆ Simple and Portable ☺
  - ◆ Less control over TCP ☹
- *Flexibility versus Functionality*

# Design Issues: *Threading Model*

---

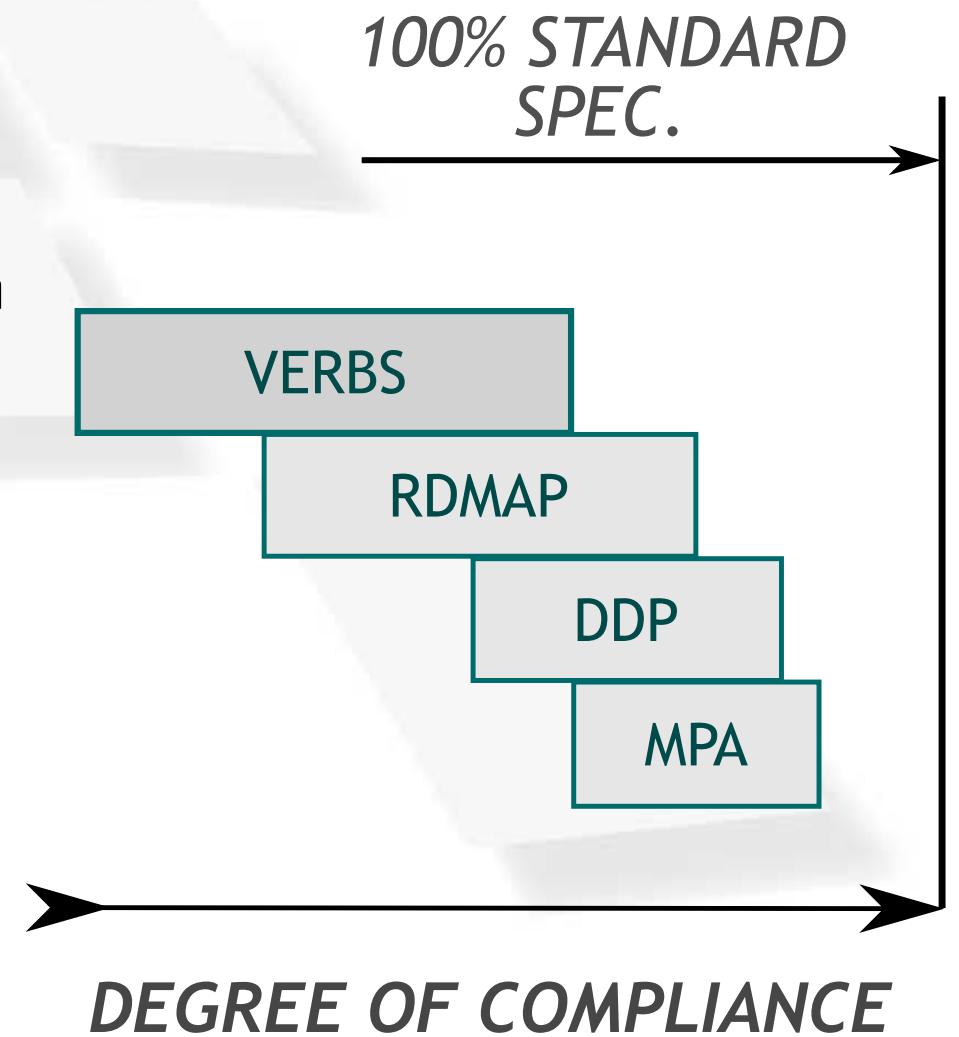
- Multi-threading
  - ◆ Mimic Hardware
  - ◆ Non-blocking, asynchronous
  - ◆ Complexity ☹
- Single Threaded model
  - ◆ Simultaneous Send Recv not possible
  - ◆ Progress Function for Recv completion
  - ◆ Simple to implement ☺
- *Simplicity versus Performance*

# Design Issues: User Kernel Interface

- Memory Copy
  - ◆ `copy_from_user` and `copy_to_user`
- Memory Pinning
  - ◆ Pre-registration of application buffers
  - ◆ `kmap` and `kunmap`
  - ◆ Book-keeping using reference counting
  - ◆ Overhead in case of 32-bit
- *Copy versus Kernel mapping*

# About the code base

- iWarp software stack works against Ammasso 1100 RNIC
- CRC and Markers: switched on and off
- 20,000 lines of ANSI C code (user and kernel)
- Linux 2.6 kernel
- 32-bit and 64-bit support

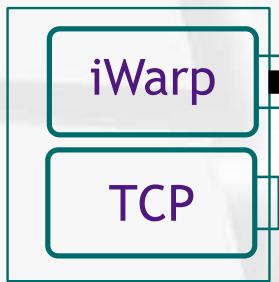


# Experimental Setup

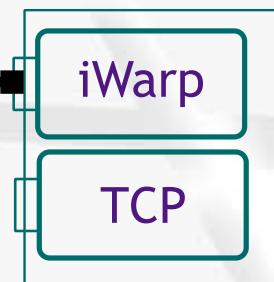
- 71 node cluster with 41 Ammasso 1100 RNIC cards
  - ◆ Beta cards with FPGA-based IP
  - ◆ RDMA data-path and TCP data-path
- Dual Opteron 250 processors
  - ◆ One processor disabled for utilization tests
- 2GB RAM, 80GB SATA drives
- 2 Tigon Gigabit Ethernet NICs
- Tyan S2891 Motherboard
- 2 SMC switches
  - ◆ Switches introduce  $2.8 \mu s$  latency

# Test Configurations

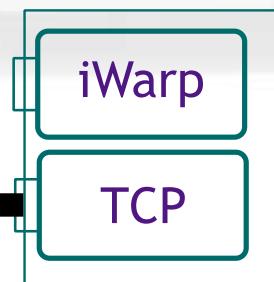
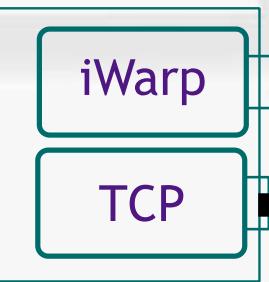
NODE 1



NODE 2

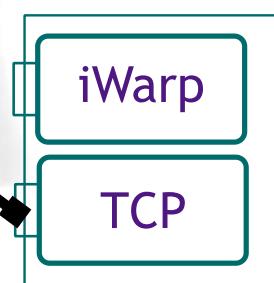
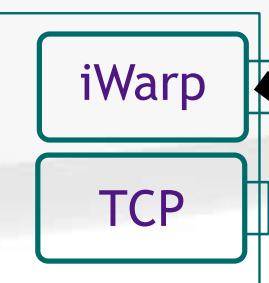


HW ↔ HW



TCP ↔ TCP

SW ↔ SW



HW ↔ SW

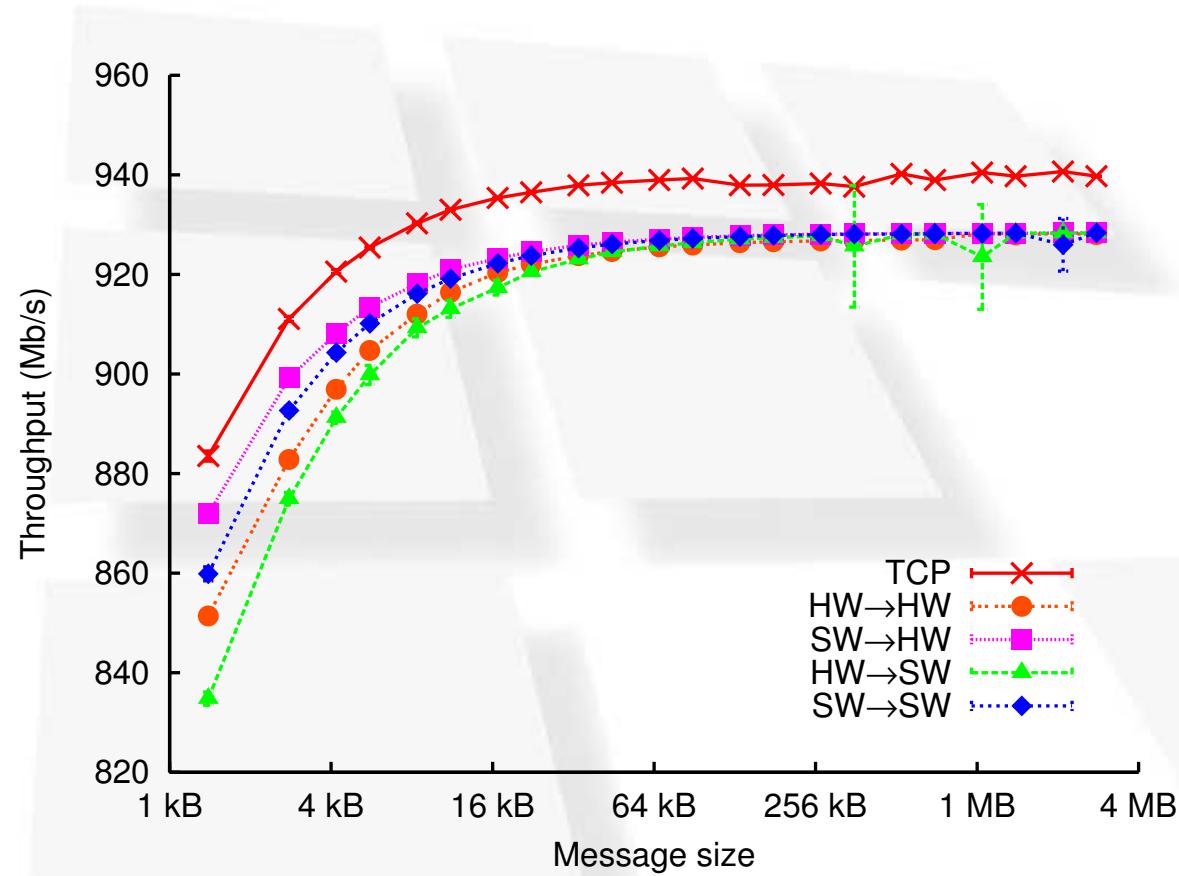
# Latency

	4 byte messages	64 kB messages
hw-hw	$16.1 \pm 0.3$	$614.2 \pm 3.3$
ksw-hw	$18.7 \pm 0.2$	$619.7 \pm 1.2$
tcp-tcp	$16.9 \pm 0.2$	$594.8 \pm 18.9$

Table 1: Latency overview ( $\mu s$ ).

- Latency: 1/2-way ping-pong delay
- Back-to-back: bypass switch
- Small overhead

# Throughput



- Sender and Receiver
- TCP > 10 Mbps

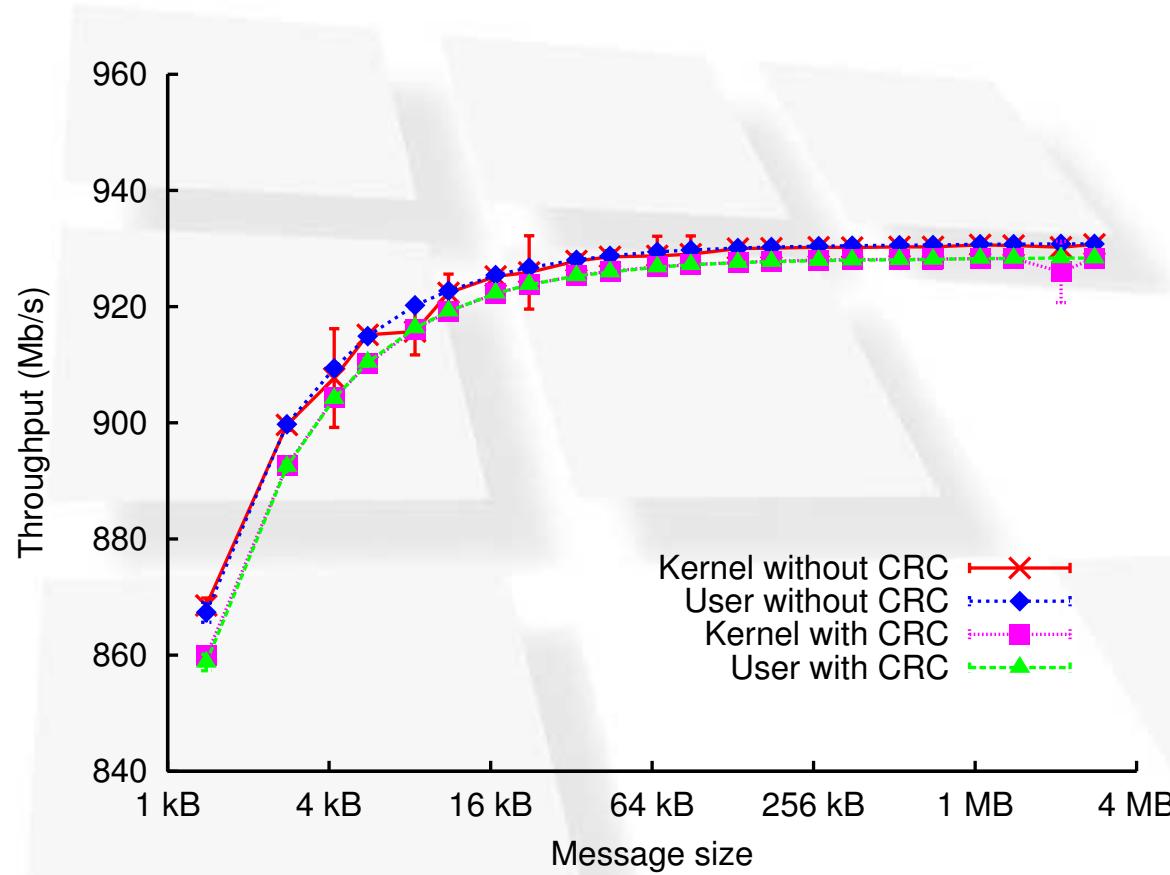
# Latency: Kernel v/s User

	4 B messages	64 kB messages
kernel with CRC	$20.3 \pm 0.2$	$615.5 \pm 1.2$
user with CRC	$19.6 \pm 0.2$	$612.3 \pm 1.9$
kernel without CRC	$20.1 \pm 0.2$	$604.5 \pm 0.8$
user without CRC	$19.5 \pm 0.2$	$602.7 \pm 0.8$

Table 2: User vs kernel space latency ( $\mu s$ ).

- CQ in kernel
- kmap/kunmap **overhead**

# Throughput: Kernel v/s User



- CRC takes away 8 Mbps
- Kernel and User space similar

# CPU utilization Hardware

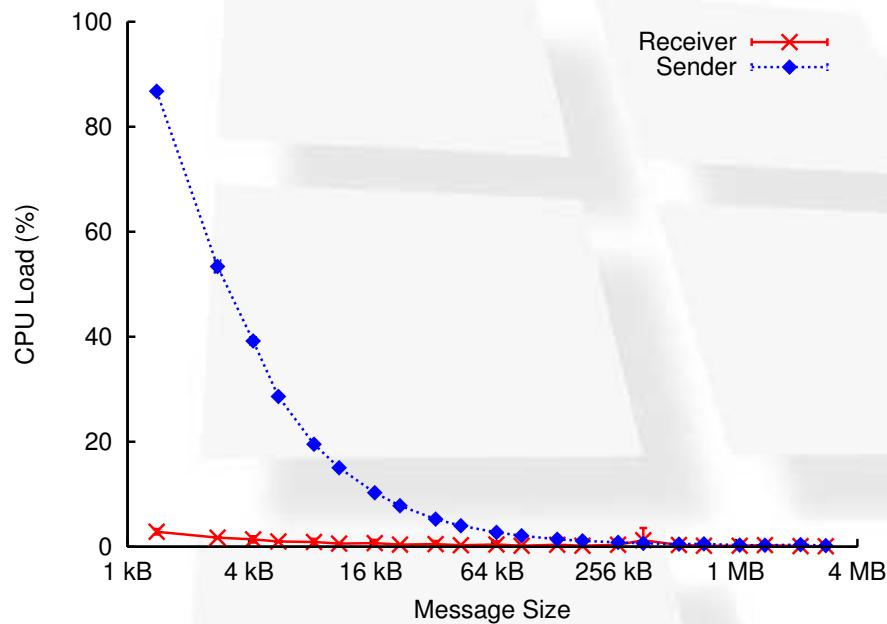


Figure 1: Hardware  $\Leftrightarrow$  Hardware

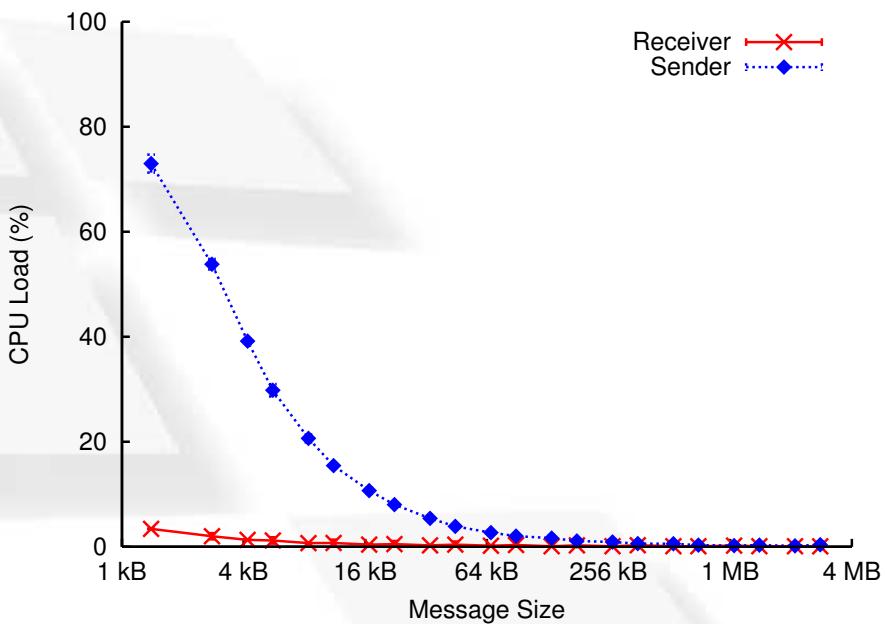


Figure 2: Hardware  $\Leftrightarrow$  Software

- Subtractive method
- Hardware and Software *almost* identical

# CPU utilization TCP and Software

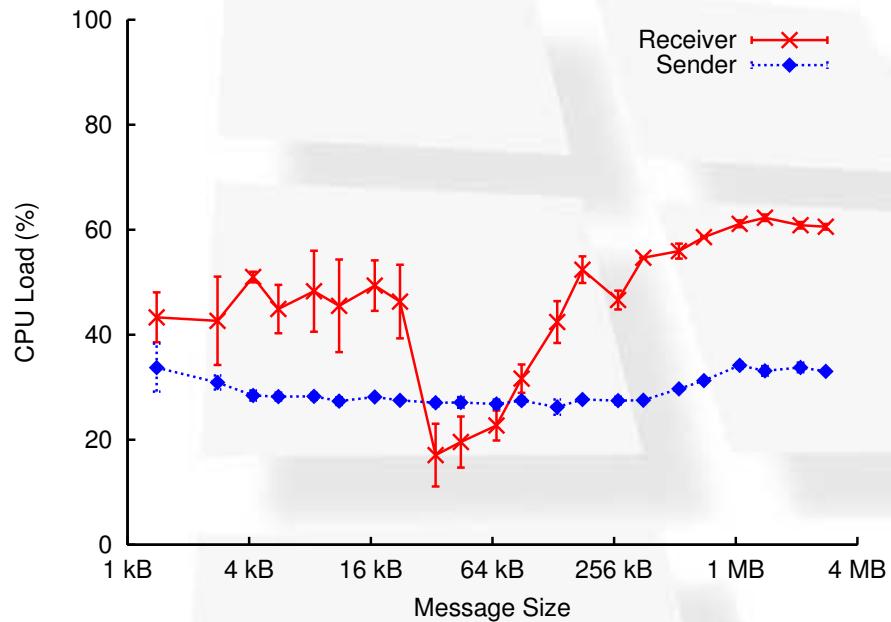


Figure 3: TCP  $\leftrightarrow$  TCP

- TCP costly than Hardware iWarp
- Software is CPU intensive: *CRC*

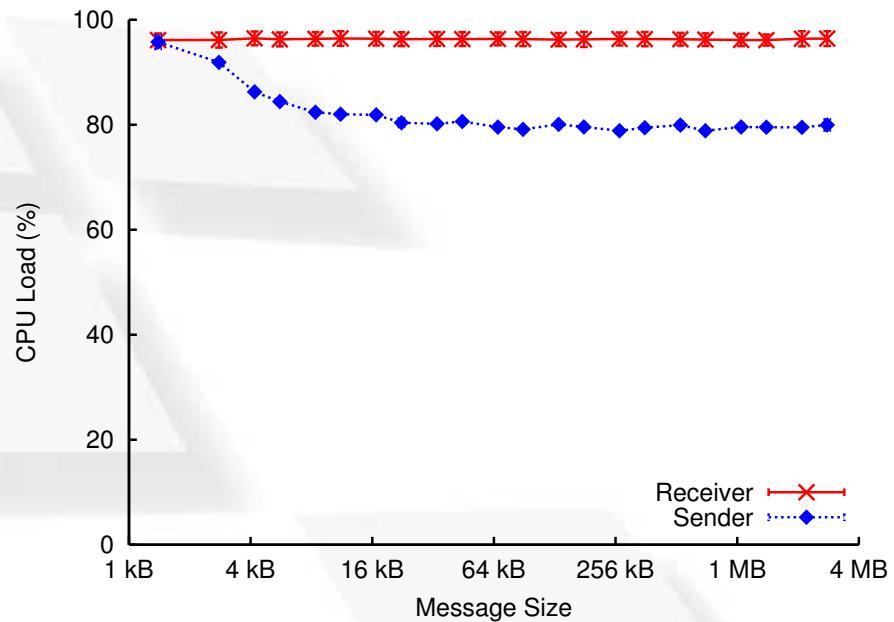
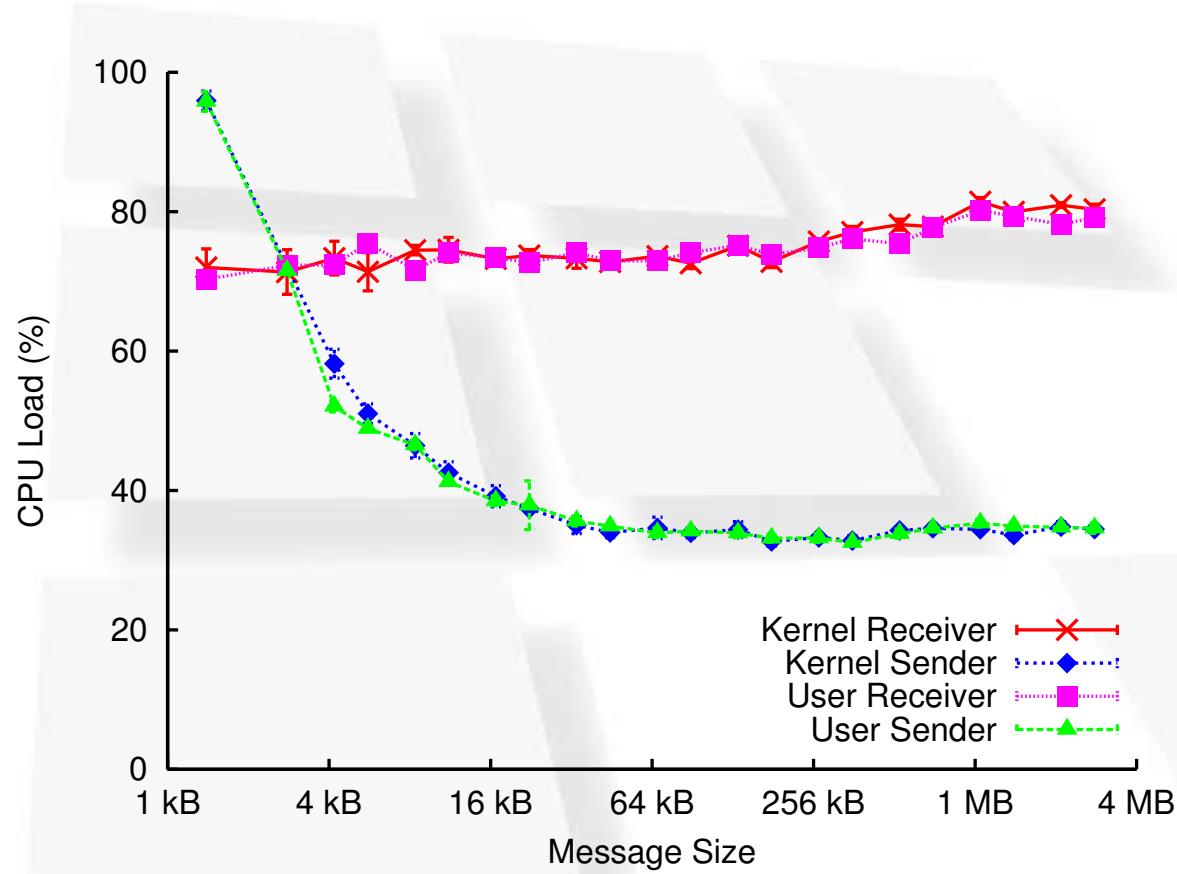


Figure 4: Software  $\leftrightarrow$  Software

# CPU Utilization without CRC



- 20% for Reciever and 40% Sender load due to CRC
- The loads in range of TCP

# Related Work

---

- User-space software iWarp
- Sockets-based iWarp
- DAT Collaborative, OpenFabrics

# Future Work

---

- Multithreaded stack
- Integrating MPA with TCP
- Porting kernel space clients
  - ◆ Verbs/API for kernel resident clients
- iSER/SRP extensions

# Conclusions

---

- Demonstrated interoperability with Hardware iWarp
- Demonstrated single-sided acceleration capability
- Unlocked Software iWarp for kernel-resident clients
- Software iWarp is logical step before full deployment

# Software Availability

---

- Thanks to Sandia for funding

*[http://www.osc.edu/research/network\\_file/projects/iwarp/index.shtml](http://www.osc.edu/research/network_file/projects/iwarp/index.shtml)*